
Test-Time Spectrum-Aware Latent Steering for Zero-Shot Generalization in Vision-Language Models

Konstantinos M. Dafnis*

Department of Computer Science
Rutgers University

Dimitris N. Metaxas

Department of Computer Science
Rutgers University

Abstract

Vision-Language Models (VLMs) excel at zero-shot inference but often degrade under test-time domain shifts. For this reason, episodic test-time adaptation strategies have recently emerged as powerful techniques for adapting VLMs to a single unlabeled image. However, existing adaptation strategies, such as test-time prompt tuning, typically require backpropagating through large encoder weights or altering core model components. In this work, we introduce **Spectrum-Aware Test-Time Steering (STS)**, a *lightweight adaptation framework* that extracts a spectral subspace from the textual embeddings to define principal semantic directions and learns to steer latent representations in a spectrum-aware manner by adapting a small number of per-sample shift parameters to minimize entropy across augmented views. STS operates entirely at inference in the latent space, without backpropagation through or modification of the frozen encoders. Building on standard evaluation protocols, our comprehensive experiments demonstrate that STS largely surpasses or compares favorably against state-of-the-art test-time adaptation methods, while introducing only a handful of additional parameters and achieving inference speeds up to 8× faster with a 12× smaller memory footprint than conventional test-time prompt tuning. The code is available at <https://github.com/kdafnis/STS>.

1 Introduction

Vision-Language Models (VLMs), such as CLIP [30], have marked a paradigm shift in artificial intelligence, demonstrating remarkable zero-shot generalization capabilities across a multitude of downstream visual tasks. By learning rich joint representations from vast quantities of image-text data, these models can often perform tasks like image classification without task-specific training, relying instead on natural language prompts to define categories [32, 43]. This ability significantly reduces the need for extensive labeled datasets and the maintenance of numerous specialized models, paving the way for more versatile and scalable AI systems.

Despite their impressive zero-shot performance, the efficacy of VLMs can be substantially compromised when encountering out-of-distribution (OOD) data, where test samples exhibit characteristics different from those seen during pre-training [32]. Such distribution shifts are common in real-world applications, leading to a degradation in model reliability. To mitigate this, Test-Time Adaptation (TTA) has emerged as a crucial strategy, enabling models to dynamically adapt to unlabeled test samples on the fly, thereby enhancing robustness while preserving the benefits of zero-shot learning [36, 35, 23]. Episodic TTA, which adapts the model for each individual test sample, is particularly relevant for VLMs operating in diverse and unpredictable environments [32, 7].

Current TTA approaches for VLMs often focus on optimizing learnable components. Test-Time Prompt Tuning (TPT) [32] and its variants [39, 9] adapt textual prompts by minimizing objectives

*Correspondence to: kd703@scarletmail.rutgers.edu

like marginal entropy over augmented views of a test sample. Although effective, these methods typically require backpropagation through the large text encoders of VLMs, leading to considerable computational overhead and increased memory usage during inference [34]. Other strategies involve parameter-efficient fine-tuning techniques such as Low-Rank Adaptation (LoRA) [17, 18] applied to parts of the VLM. However, these approaches often necessitate access to and modification of the model’s internal architecture, deviating from a truly black-box paradigm and potentially limiting their applicability to proprietary models or those with fixed structures. The challenge remains to develop TTA methods that are both highly efficient and minimally invasive while effectively addressing domain shifts.

To address these limitations, we introduce Spectrum-Aware Test-Time Steering (STS), a novel TTA framework for VLMs that operates by efficiently adapting text representations within a low-dimensional subspace defined by their Singular Value Decomposition (SVD). Instead of learning prompt vectors or modifying encoder weights, STS pre-computes a semantic basis from the SVD of the initial class text embeddings. At test time, for each incoming sample, our method learns a small set of coefficients that define a shift vector within this principal SVD subspace. This shift is then applied to the initial text prototypes, effectively steering them in the joint embedding space to better align with the current visual input. This approach directly manipulates representations in the latent space in a highly targeted and parameter-efficient manner.

The core strength of STS lies in its strategy of adapting representations within a structured, low-dimensional subspace, a design motivated by the observation that embeddings derived from pre-trained deep neural networks are typically characterized by a low intrinsic dimension [2]. This implies that their essential information resides within a lower-dimensional manifold, which can be effectively identified through spectral decomposition methods such as SVD. By operating on the principal singular vectors derived from the initial text embeddings, our method explicitly leverages the inherent semantic geometry of the VLM’s text feature space. These singular vectors capture the most salient axes of variation among class concepts, providing a robust and semantically grounded basis for adaptation. Performing test-time adaptation within this constrained subspace inherently regularizes the learning process, fostering enhanced stability against noisy augmentations or idiosyncrasies of individual test samples. This targeted manipulation not only preserves the rich knowledge encoded in the frozen VLM but also ensures that adaptations are focused along directions of maximal semantic relevance, rather than allowing unconstrained shifts in the high-dimensional embedding space.

This work has four primary contributions:

- We propose STS, a novel TTA method that, to the best of our knowledge, is the first to leverage the SVD-defined latent subspace of text embeddings for efficient and effective adaptation of VLMs to unlabeled test data.
- Our method exhibits significant computational advantages: it avoids backpropagation through the VLM encoders and adapts only a minimal number of parameters (the SVD subspace coefficients), leading to substantially lower latency and memory footprint compared to conventional prompt tuning techniques.
- STS operates as a black-box adaptation mechanism, treating the VLM encoders as fixed feature extractors without requiring knowledge of or modifications to their internal architectures. This makes our approach broadly applicable and non-invasive.
- Extensive experiments on several benchmark datasets for natural distribution shifts and cross-dataset generalization demonstrate that STS achieves state-of-the-art performance, enhancing the zero-shot capabilities of VLMs efficiently and effectively.

2 Related work

Vision-Language Models. Pre-trained on extensive image-text datasets via self-supervised learning, vision-language models (VLMs) like CLIP [30] and ALIGN [19] exhibit impressive generalization abilities. For instance, CLIP’s exceptional zero-shot performance largely stems from the scale and variety of its training data. However, effectively adapting these models to specific downstream tasks, especially in data-scarce scenarios, continues to pose significant challenges. Efforts to improve the transferability of vision-language models have led to the use of prompt tuning techniques, including CoOp [43], CoCoOp [42], and MaPLe [21], and adapter-based methods, such as Tip-Adapter [40]

and CLIP-Adapter [10]. However, a prevalent assumption in these techniques is the accessibility of labeled data from the target domain, a condition that is frequently incompatible with the requirements for rapid deployment in real-world applications. Therefore, we focus on test-time adaptation, defined as the challenge of a model adapting to a target domain using only the unlabeled test instances, with no access to any training data or ground-truth labels from that specific domain.

Test-Time Adaptation. Test-Time Adaptation (TTA) aims to improve model robustness and generalization by adapting a pre-trained model to unlabeled test data encountered during inference [35, 23, 36, 11, 26, 4, 32]. In the context of VLMs, several TTA strategies have emerged.

A dominant paradigm involves tuning the learnable prompt vectors at test time. Test-Time Prompt Tuning (TPT) [32] pioneered this by optimizing textual prompts for each test sample to minimize the entropy of predictions over augmented views. Subsequent works have built upon TPT, such as DiffTPT [9], which employs diffusion models for more diverse augmentations, and C-TPT [39], which focuses on improving model calibration during TTA. Although these methods adapt VLMs without labeled data, they generally incur significant computational costs and memory overhead due to the need for backpropagation through the large VLM encoders to update the prompt parameters.

To address the efficiency concerns of prompt tuning, training-free approaches have been proposed. Most of these methods operate in an on-line streaming scenario, using memory banks to retain information from previously seen test inputs [41, 20]. However, such methods suffer from critical drawbacks that limit their applicability to real-life scenarios. First, maintaining memory banks significantly increases memory consumption, which becomes prohibitive for resource-constrained devices or large-scale deployments. Second, their efficacy hinges on the assumption of a well-distributed stream of data, an unrealistic expectation in practice, as imbalanced or nonstationary test distributions may prevent the memory bank from accumulating sufficient or representative samples within a reasonable time-frame. This dependency renders them unreliable for time-sensitive applications or scenarios with sparse or bursty data streams. Furthermore, memory-based methods risk performance degradation when test samples arrive in biased sequences, as stored information may reflect transient patterns rather than meaningful statistical trends. Our STS method, while involving a lightweight optimization step, shares the goal of minimizing encoder backpropagation and architectural changes.

Other approaches explore parameter-efficient fine-tuning (PEFT) techniques at test time. For example, TTL [18] adapts LoRA [17] parameters within the VLM’s attention layers. Although more efficient than full fine-tuning, such methods still typically require modification of the underlying model architecture, differing from black-box approaches where the VLM encoders are treated as fixed.

Directly adapting or modulating representations in the latent embedding space offers an alternative to prompt tuning or architectural modifications. Test-Time Prototype Shifting (TPS) [34] proposes learning shift vectors for pre-computed class prototypes directly in the embedding space, thereby avoiding backpropagation through encoders and achieving significant efficiency gains. This is conceptually related to our work. However, TPS learns unconstrained shift vectors for each class prototype. Our proposed STS method advances the idea of latent space adaptation by introducing a spectrum-aware mechanism. Instead of learning arbitrary shifts, STS learns compact coefficients that define shifts along principal semantic axes derived from the SVD of text embeddings. This constrains the adaptation to a low-dimensional, semantically meaningful subspace, aiming for both efficiency and effectiveness.

3 Method

In this section, we first provide preliminary details on Vision-Language Models (VLMs) and the test-time adaptation setting. Subsequently, we introduce our proposed Spectrum-Aware Test-Time Steering (STS) framework. We detail its components for identifying a principal spectral subspace from text embeddings, the mechanism for latent steering via a shared, learnable coefficient vector operating within this subspace, and the test-time optimization objective.

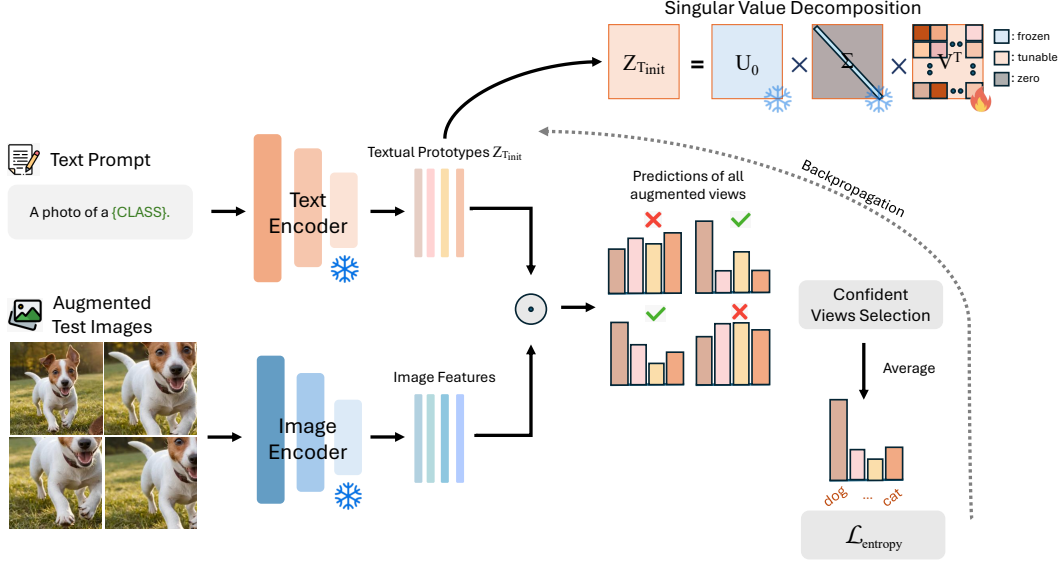


Figure 1: Overview of our proposed STS framework. Given text and image inputs, encoders $\mathcal{E}_t(\cdot)$ and $\mathcal{E}_v(\cdot)$ extract text embeddings/prototypes, and visual embeddings. A probability distribution $\mathbb{P}_{CLIP}(y = y_c | X_{\text{test}})$ is computed based on these embeddings. Then we perform a refinement step of test-time adaptation, where we tune the learnable low-dimensional coefficients to generate a small steering to the text prototypes to close the gap between the source and target distributions. Marginal entropy of the CLIP similarities of the shifted embeddings and the class prototypes is minimized.

3.1 Preliminaries

Vision-Language Models (VLMs). Pre-trained VLMs, such as CLIP [30], comprise a visual encoder $\mathcal{E}_v(\cdot)$ and a textual encoder $\mathcal{E}_t(\cdot)$, which map images x and text descriptions t into a shared D -dimensional embedding space. For a C -class zero-shot image classification task, a set of initial text prototypes, $Z_{T_{init}} = \{(z_{T_{init}})_c\}_{c=1}^C \in \mathbb{R}^{C \times D}$, is derived by encoding class names, typically using prompt templates (e.g., "a photo of a {class name}"). Given an input image x , its visual embedding is $z_v = \mathcal{E}_v(x) \in \mathbb{R}^D$. The predicted probability for class c is:

$$p(y = c | x, Z_{T_{init}}) = \frac{\exp(\text{sim}(z_v, (z_{T_{init}})_c) / \tau)}{\sum_{i=1}^C \exp(\text{sim}(z_v, (z_{T_{init}})_i) / \tau)}, \quad (1)$$

where $\text{sim}(\cdot, \cdot)$ denotes cosine similarity, and τ is CLIP's learned temperature.

Test-Time Adaptation (TTA). Episodic TTA seeks to adapt a model f for each unlabeled test sample x_{test} by optimizing a small set of parameters θ_{TTA} using an objective \mathcal{L}_{TTA} derived from x_{test} (typically via augmentations). The adapted model $f(\cdot; \theta_{\text{TTA}}^*)$ is then used for prediction, and θ_{TTA} is reset for subsequent samples.

3.2 Spectrum-Aware Test-Time Steering (STS)

STS adapts VLMs at test time by *learning to steer* the initial text prototypes. This steering is enacted by a single, shared vector of learnable coefficients that operates within a low-dimensional semantic subspace. This subspace is critically derived from the spectral properties (Singular Value Decomposition) of the initial text embeddings, ensuring adaptations are both efficient and aligned with principal semantic variations.

3.2.1 Spectral Subspace Identification from Text Prototypes

The core of STS lies in identifying a robust, low-dimensional subspace from the initial text prototypes $Z_{T_{init}} \in \mathbb{R}^{C \times D}$ to guide the adaptation. High-dimensional embedding spaces, while expressive, can be susceptible to noise and may contain redundant information for specific adaptation tasks. Pre-trained

features from deep neural networks often exhibit a low intrinsic dimensionality [2], implying that their essential information can be effectively captured within a lower-dimensional manifold. By projecting the adaptation process onto such a manifold, defined by the principal spectral components, STS aims to achieve more stable, generalizable, and semantically meaningful adaptations.

To this end, we perform a Singular Value Decomposition (SVD) on the initial text prototypes $Z_{T_{\text{init}}}$. Using the reduced SVD:

$$Z_{T_{\text{init}}} = U_T S_T V_T^\top, \quad (2)$$

where $U_T \in \mathbb{R}^{C \times k'}$ contains the left singular vectors, $S_T \in \mathbb{R}^{k' \times k'}$ is a diagonal matrix of $k' = \min(C, D)$ singular values $s_1 \geq s_2 \geq \dots \geq s_{k'} \geq 0$, and $V_T^\top \in \mathbb{R}^{k' \times D}$ has rows corresponding to the right singular vectors. The columns of $V_T \in \mathbb{R}^{D \times k'}$ (i.e., the right singular vectors of $Z_{T_{\text{init}}}$) represent an orthonormal basis for the subspace capturing the principal directions of variance in the text prototype data. These directions correspond to the most significant semantic axes that differentiate the class concepts as represented by $Z_{T_{\text{init}}}$.

We select the top k_t right singular vectors from V_T (associated with the k_t largest singular values) to form our textual adaptation basis $B_T = [v_1, v_2, \dots, v_{k_t}] \in \mathbb{R}^{D \times k_t}$. The choice of $k_t \ll D$ is pivotal for efficiency and robustness. A small k_t focuses adaptation on the most dominant semantic variations, potentially filtering out noise associated with higher-order components and leveraging the aforementioned low intrinsic rank of deep features [2]. Empirically, as we see in Figure 2, a small k_t often captures the vast majority (e.g., >90%) of the total variance (sum of squared singular values, or "energy") of $Z_{T_{\text{init}}}$. Thus, for automatic and principled selection of k_t , we employ the optimal hard thresholding strategy proposed by Gavish and Donoho [12]. This method determines an optimal singular value threshold ω^* based on the aspect ratio of the matrix $Z_{T_{\text{init}}}$ (i.e., C/D) and the median of its singular values. The rank k_t is then the count of singular values s_i such that $s_i > \omega^*$. This pre-computed basis B_T defines the k_t -dimensional spectral subspace for our test-time steering.

3.2.2 Latent Steering via Subspace Coefficients

For each incoming test image x_{test} , STS learns a *single, shared* vector of k_t learnable coefficients $\gamma \in \mathbb{R}^{k_t}$. These coefficients determine the magnitude and direction of the text prototype steering along each of the k_t basis vectors in B_T . The steering vector (shift) $\Delta z_T \in \mathbb{R}^D$, which is applied to all class prototypes, is reconstructed from its k_t -dimensional representation γ :

$$\Delta z_T = B_T \gamma. \quad (3)$$

The adapted text prototype $(z_{T_{\text{adapted}}})_c$ for each class c is then obtained by:

$$(z_{T_{\text{adapted}}})_c = \text{normalize}((z_{T_{\text{init}}})_c + \Delta z_T). \quad (4)$$

The only parameters learned at test time are the k_t coefficients in γ .

3.2.3 Test-Time Optimization Objective

The shared steering coefficients γ are optimized for each x_{test} using an unsupervised objective based on prediction consistency over N augmented views of x_{test} , denoted $\{x^{(j)}\}_{j=1}^N$. Visual embeddings $Z_V = \{z_v^{(j)} = \mathcal{E}_v(x^{(j)})\}_{j=1}^N$ are extracted using the frozen \mathcal{E}_v .

Confidence Filtering. Following prior TTA works [32, 7], views are filtered based on prediction confidence using the initial (unadapted) text prototypes $Z_{T_{\text{init}}}$. Logits $L_{\text{init}}^{(j)}$ and probabilities $P_{\text{init}}^{(j)}$ are computed per view. Views with prediction entropy $H(P_{\text{init}}^{(j)})$ falling within the top- ρ percentile of confidence (lowest entropy) are retained, forming $Z_{V_{\text{filt}}} = \{z_v^{(j')}\}$ of size N_{filt} .

Marginal Entropy Minimization. For $z_v^{(j')} \in Z_{V_{\text{filt}}}$, logits with current adapted text prototypes $Z_{T_{\text{adapted}}}$ (from Eq. 4 using current γ) are: $L_{\text{adapted}}^{(j')}(c) = \text{sim}(z_v^{(j')}, (Z_{T_{\text{adapted}}})_c) / \tau$. The marginal probability distribution \bar{P}_{adapted} is:

$$\bar{P}_{\text{adapted}}(c) = \frac{1}{N_{\text{filt}}} \sum_{j'=1}^{N_{\text{filt}}} \text{softmax}_c(L_{\text{adapted}}^{(j')}(c)). \quad (5)$$

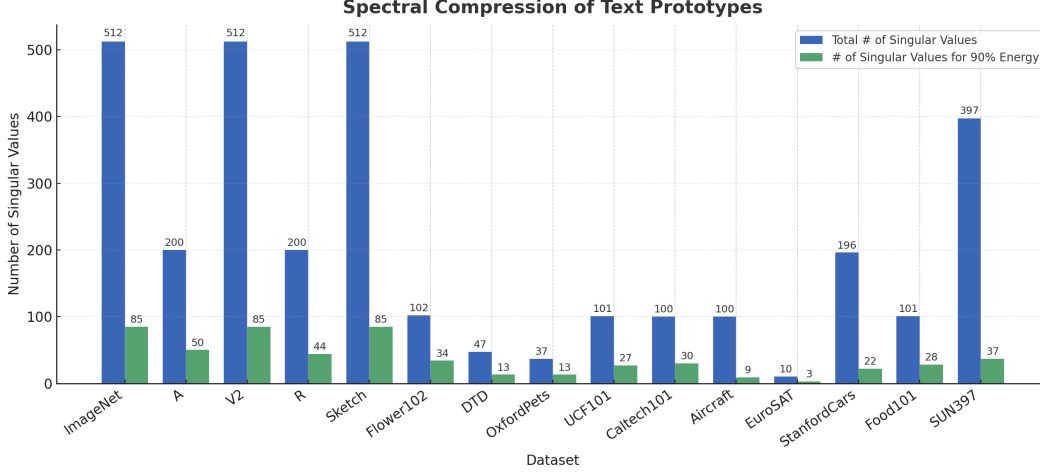


Figure 2: Most spectral energy in CLIP text prototypes is captured by a small subset of singular values, highlighting strong low-rank structure across datasets.

The primary objective is to minimize the Shannon entropy of \bar{P}_{adapted} :

$$\mathcal{L}_{\text{ent}} = H(\bar{P}_{\text{adapted}}) = - \sum_{c=1}^C \bar{P}_{\text{adapted}}(c) \log \bar{P}_{\text{adapted}}(c). \quad (6)$$

An L2 regularization term is added for Δz_T : $\mathcal{L}_{\text{reg}} = \lambda_R \|\Delta z_T\|_2$. The total loss is:

$$\mathcal{L}_{\text{STS}} = \mathcal{L}_{\text{ent}} + \mathcal{L}_{\text{reg}}. \quad (7)$$

The coefficients γ are initialized to zeros and optimized to minimize \mathcal{L}_{STS} .

3.2.4 Inference with Adapted Prototypes

After optimization yielding γ^* , the final adapted text prototypes $Z_{T_{\text{final}}}$ are computed. The final class prediction \hat{y} for x_{test} is:

$$\hat{y} = \operatorname{argmax}_c \left(\frac{1}{N_{\text{filt}}} \sum_{j'=1}^{N_{\text{filt}}} \operatorname{softmax}_c \left(\operatorname{sim}(z_v^{(j')}, (Z_{T_{\text{final}}})_c) / \tau \right) \right). \quad (8)$$

4 Experiments and Results

We conduct experiments on a diverse range of benchmark datasets to assess the performance and robustness of our method, specifically testing its out-of-domain generalization across different domains.

4.1 Experimental Setup

Datasets. We conduct a comprehensive evaluation of our method across a diverse set of benchmark datasets, with a particular focus on out-of-domain generalization. To assess the model’s ability to handle distribution shifts, we utilize several ImageNet variants, including ImageNet-A [16], ImageNet-V2 [31], ImageNet-R [14], and ImageNet-Sketch (also referred to as ImageNet-K) [37]. These datasets serve as established Out-of-Distribution (OOD) benchmarks for ImageNet, enabling a rigorous assessment of the model’s robustness under varying conditions and data distributions.

For Fine-grained Classification (also referred to as "Cross-Datasets Generalization in previous works), in line with [32], we include Flowers102 [27], DTD [5], Pets [29], UCF [33], and Caltech101 [8]. These datasets facilitate the evaluation of the model’s capacity to distinguish fine-grained

Table 1: Comparison of top-1 accuracy (%) across ImageNet and its OOD variants. The best results in each section are highlighted in **bold**. Underline indicates second-best.

Method	ImageNet	A	V2	R	Sketch	Average	OOD Average
CLIP-ViT-B/16							
Zero-Shot [30]	66.73	47.87	60.86	73.98	46.09	59.11	57.2
Ensemble [28]	68.34	49.89	61.88	77.65	48.24	61.20	59.42
CoOp [43]	71.51	49.71	64.20	75.21	47.99	61.72	59.28
TPT [32]	<u>68.97</u>	54.39	63.37	77.07	48.01	62.36	60.71
DiffTPT [9]	70.30	55.68	65.10	75.00	46.80	62.28	60.52
C-TPT [39]	68.53	51.14	62.13	75.66	47.37	60.97	59.08
TPS [34]	67.96	57.46	62.95	74.90	46.03	61.86	60.34
STS (Ours)	68.85	<u>61.23</u>	64.15	<u>77.13</u>	<u>48.06</u>	<u>63.88</u>	<u>62.64</u>
STS _{Ensemble}	70.81	64.29	<u>64.82</u>	80.53	50.19	66.13	64.96
MaPLe							
Zero-Shot [21]	-	50.90	64.07	76.98	<u>49.15</u>	-	60.28
TPT [32]	-	<u>58.08</u>	<u>64.87</u>	<u>78.12</u>	48.16	-	<u>62.31</u>
STS (Ours)	-	64.83	66.49	79.43	50.62	-	65.34

variations among visually similar classes. Furthermore, to assess the model’s adaptability across diverse domains, we incorporate Aircraft [25], EuroSAT [13], Cars [22], Food [3], and SUN397 [38], encompassing a broad spectrum of data modalities, including aerial and satellite imagery, object-centric datasets, and scene-centric environments. For all datasets, we utilize the test splits defined by Zhou et al. [43], adhering to the common evaluation protocol.

Implementation Details. Following TPT [32], we generate 63 augmented versions of a test image using *only* random resized crops and horizontal flips *for all datasets*, unlike prior methods that use task-specific augmentations such as AugMix, resulting in a batch of 64 images including the original input. To identify high-confidence samples, we select the 10% of batch samples with the lowest entropy and compute the marginal entropy based on their predicted probability distributions. The learnable vector is initialized to zero and optimized for a single step using the AdamW [24] optimizer with a learning rate of 5e-3 across all datasets. In our method, each class prototype is initialized using the hand-crafted prompt, “a photo of a {CLASS}.” All experiments are conducted on a single NVIDIA RTX8000 GPU with 45GB of memory. The presented results are an average taken over three distinct random seeds. Top-1 accuracy is reported in all tables, unless explicitly indicated otherwise.

Baselines. We evaluate our method against zero-shot and test-time adaptation (TTA) baselines that utilize CLIP ViT-B/16 as the backbone. The TTA methods considered include TPT [32], which performs text-prompt tuning; DiffTPT [9], a variant of TPT that employs diffusion models to augment visual training data; TPS [34], which optimizes a shift vector for each class prototype; and C-TPT [39], an extension of TPT that improves model calibration by selecting prompts based on the dispersion of textual embeddings. To ensure a fair comparison, we reproduce TPT, TPS, and C-TPT on our system using a single update step and the same backbone architecture. For DiffTPT, we report results from [9]. It is important to note that the DiffTPT study evaluates performance on a subset of each dataset that contain only 1,000 test samples, which may introduce potential variability in the reported results.

Textual Prompts. When *Ensemble* is specified, we do not use dataset-specific templates. In contrast, we use the set of 7 generic templates highlighted in the official CLIP repository [28] across all datasets.

4.2 STS Results

Natural Distribution Shifts. Table 1 presents the top-1 accuracy of our method, comparing it against zero-shot and test-time adaptation (TTA) baselines using CLIP on ImageNet and its out-

Table 2: Performance comparisons on fine-grained classification. The best results in each section are highlighted in **bold**. Underline indicates second-best.

Method	Flowers102	DTD	OxfordPets	UCF101	Caltech101	Aircraft	EuroSAT	StanfordCars	Food101	SUN397	Average
CLIP-ViT-B/16											
Zero-Shot [30]	67.44	44.27	88.25	65.13	93.35	23.67	42.01	65.48	83.65	62.59	63.58
Ensemble [28]	66.99	45.04	86.92	65.16	93.55	23.22	50.42	66.11	82.86	65.63	64.59
CoOp [43]	68.71	41.92	89.14	66.55	93.70	18.47	46.39	64.51	85.30	64.15	63.88
TPT [32]	68.98	47.16	87.07	67.89	94.19	22.85	43.01	66.55	84.67	<u>65.47</u>	<u>64.78</u>
C-TPT [39]	69.88	45.54	87.96	65.19	93.39	24.13	38.43	65.26	82.60	63.38	63.58
TTL [18]	67.32	45.92	86.78	<u>67.80</u>	93.23	24.00	36.52	65.95	84.40	64.02	63.59
TPS [34]	66.14	45.49	86.56	66.53	93.60	24.01	37.85	66.93	82.96	64.85	63.49
STS (Ours)	66.10	46.02	86.69	66.52	93.72	24.57	38.26	<u>67.17</u>	<u>84.72</u>	64.79	63.86
STS _{Ensemble}	67.16	<u>46.87</u>	<u>87.11</u>	67.14	94.20	<u>24.21</u>	43.80	68.16	85.15	66.79	65.06
MaPLe											
Zero-Shot [21]	<u>72.23</u>	46.49	<u>90.49</u>	68.69	93.53	<u>24.74</u>	48.06	65.57	86.20	67.01	66.30
TPT [32]	72.37	45.87	90.72	69.19	<u>93.59</u>	24.70	47.80	<u>66.50</u>	86.64	67.54	66.49
STS (Ours)	70.70	47.60	90.00	<u>68.94</u>	94.02	25.44	40.83	68.32	<u>86.56</u>	<u>67.26</u>	<u>65.97</u>

of-distribution (OOD) variants. The results demonstrate that steering the text prototypes with our learnable vector, leads to a substantial performance improvement. Specifically, our approach achieves an average OOD performance gain of 7.76% over the zero-shot CLIP baseline and a 4.23% improvement over standard TPT across OOD datasets.

Furthermore, Table 1 shows that simply learning a shift vector for per-class prototypes in TPS results in a slight performance drop (< 0.6 points) compared to TPT, highlighting the limitations of TPS in effectively aligning text prototypes with out-of-distribution visual embeddings. Additionally, our method is significantly more efficient, as STS runs 8 times faster than TPT (see Table 3) while still achieving superior performance. These substantial speed-ups make our approach highly practical for real-world applications.

Table 3: Efficiency comparison on ImageNet. We report the testing time per sample, the memory usage, the accuracy, and the performance gains compared to zero-shot CLIP.

Method	Testing Time (s)	Memory (GB)	Accuracy	Gain
Zero-Shot	0.02	0.83	66.73	-
TPT	0.75	17.6	68.97	+2.24
STS _{Ensemble}	0.09	1.4	70.81	+4.08

Fine-grained Classification. We further assess the generalization capabilities of STS across ten diverse image classification datasets, with results presented in Table 2. This benchmark evaluates the model’s ability to adapt to datasets that may differ significantly in domain and class composition from the VLM’s pre-training data.

In this challenging scenario, our STS_{Ensemble} variant, which leverages 7 generic CLIP templates, achieves the highest average accuracy of 65.06% among all methods compared. This result surpasses the standard TPT [32] (average 64.78%) and demonstrates the effectiveness of combining STS with prompt ensembling. The standard STS (using a single "a photo of a {CLASS}" prompt) achieves an average accuracy of 63.86%, which is competitive and outperforms the Zero-Shot CLIP baseline (63.58%). It also surpasses other TTA methods such as C-TPT [39] (63.58%), TTL [18] (63.59%), and TPS [34] (63.49%) on average. Standard STS particularly excels on datasets like Aircraft (24.57%, best in its group), StanfordCars (67.17%, second best), and Food101 (84.72%, second best). The STS_{Ensemble} variant shows broad strength, achieving the best results on Caltech101 (94.20%), EuroSAT (43.80%), StanfordCars (68.16%), Food101 (85.15%), and SUN397 (66.79%). This indicates that while standard TPT is a strong baseline, STS, especially when combined with prompt ensembling, offers a more robust generalization across these diverse datasets.

Overall, STS demonstrates robust and often superior performance in fine-grained classification. The STS_{Ensemble} variant, in particular, sets a new state-of-the-art average across the CLIP-ViT-B/16 backbone experiments. Even with stronger MaPLe [21] initializations, STS provides a more effective adaptation than TPT. These results, combined with STS’s significant advantages in parameter

efficiency and computational speed (detailed in Table 3), underscore its potential as a practical and powerful approach for real-world test-time adaptation of VLMs.

5 Analysis and ablation

We perform ablation studies to assess the effect of key design choices on performance. For consistency, all analyses use ImageNet and ImageNet-A with the ViT-B/16 backbone. Additionally, we evaluate STS on CIFAR10-C [15] to test its robustness under challenging distribution shifts.

5.1 Computational Analysis

Trainable Parameters: Test-time tuning approaches such as TPT, DiffTPT, and C-TPT adapt textual prompts using only 2048 trainable parameters, corresponding to four tokens with $d=512$. However, these methods exhibit limited generalization, achieving a Top-1 accuracy of approximately 51% to 55% on ImageNet-A. In contrast, visual adaptation techniques, such as encoder tuning and layer normalization optimization, require a larger number of trainable parameters. Meanwhile, STS effectively balances this trade-off with just a small number of trainable parameters, achieving the highest generalization performance at 61.23% while preserving model accessibility constraints. Table 3 shows the testing time per sample and the performance gain on ImageNet for STS and the TTA baseline TPT on a single RTX8000 GPU. STS is 8x faster than TPT, corresponding to an order of magnitude of computational savings in time.

5.2 Effect of STS on Different Prototypes

We study the impact of the steering vector under different prototype constructions. Specifically, we compare STS against the zero-shot CLIP using the standard prompt "*a photo of a {CLASS}*" and an ensemble of seven generic hand-crafted templates from the official CLIP repository. While the design of TPT does not support the use of text ensembles, STS integrates them seamlessly; we denote this variant as $\text{STS}_{\text{Ensemble}}$. As shown in Tables 1 and 2, adding these generic prompts further improves STS, surpassing TPT without relying on dataset-specific templates.

Furthermore, we evaluate our method using a MaPLE [21] initialization, where the MaPLE prompts are learned on ImageNet in a 16-shot setup following [1]. In this evaluation, we also report results for TPT applied on top of MaPLE, as in [1], which we refer to as MaPLE+TPT. Although MaPLE+TPT performs better than previous methods that rely on hand-crafted prompts, our STS method notably outperforms MaPLE+TPT on most datasets. This demonstrates that the adaptive mechanism of STS provides consistent advantages even when initialized with optimized textual and visual prompts.

Natural Distribution Shifts. Under natural distribution shifts, our STS method consistently outperforms TPT even when the baseline initialization is MaPLE, achieving an average improvement of +3.03%. We omit evaluation on ImageNet in this group, as MaPLE uses it as the source dataset for model adaptation, making the comparison unfair. For completeness, Zero-Shot MaPLE attains 70.72% accuracy on ImageNet, which further improves to 72.72% when adapted with STS (+2.0%). STS with MaPLE (MaPLE+STS) demonstrates leading performance on several natural shift datasets, including ImageNet-A (64.83%), ImageNet-V2 (66.49%), ImageNet-Sketch (50.62%).

Fine-grained Classification. In fine-grained classification tasks, MaPLE+TPT shows a marginal average improvement of +0.52% over MaPLE+STS. However, this difference is primarily driven by performance on a single dataset, EuroSAT, while MaPLE+STS surpasses MaPLE+TPT on roughly half of the remaining datasets by a considerable margin. For the datasets where MaPLE+STS lags behind, the differences remain minimal. As discussed in [7], EuroSAT constitutes a known failure mode for many TTA methods. Its analysis suggests that the unique nature of satellite imagery demands task-specific augmentation strategies, making EuroSAT a controversial benchmark for evaluating TTA performance.

5.3 Robustness of Linear Spectrum Steering

We evaluate STS on CIFAR10-C at the highest corruption level (severity 5) following the TPT protocol (10% most confident views, learning rate 0.005, and the hand-crafted prompt "a photo of

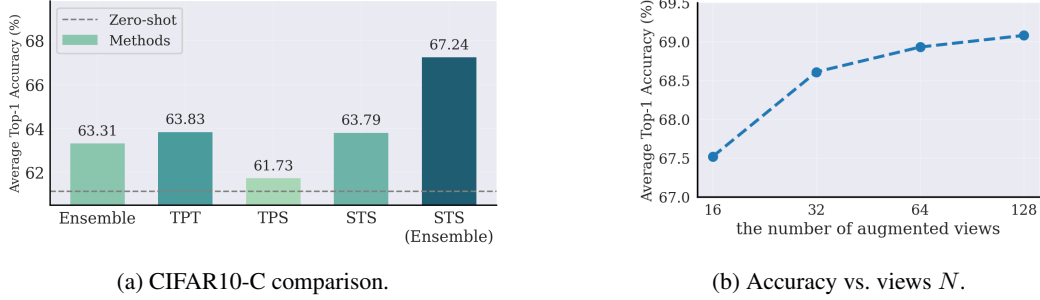


Figure 3: (a) Comparison on CIFAR10-C (severity 5). (b) Accuracy vs. number of augmented views.

a {CLASS}"). STS matches TPT within 0.05% (Figure 3a), while clearly outperforming the naive per-class shifting (TPS), confirming the effectiveness of spectrum-aware subspace steering. With the seven generic CLIP templates, STS reaches 67.24%, demonstrating strong complementarity between subspace steering and prompt ensembles. Constraining adaptation to the top- k singular vectors further stabilizes learning under severe corruptions.

5.4 Balancing Inference Efficiency and Accuracy

We analyze the impact of the number of augmented views N on STS efficiency. As shown in Figure 3b, accuracy increases with N and saturates around $N = 64$. Increasing to 128 views yields only a $\approx 0.15\%$ gain while nearly doubling time and memory. We thus adopt $N = 64$ (as in prior TPT work) to balance performance and efficiency, while remaining substantially faster than prompt-tuning baselines.

6 Limitations

While STS demonstrates significant advantages in efficiency and effectiveness for test-time adaptation, we acknowledge two limitations that warrant discussion and offer avenues for future research:

Linearity of subspace steering. The adaptation mechanism involves linear shifts (steering) within the identified SVD subspace. While this subspace captures principal linear variations, highly complex or non-linear domain shifts might not be fully addressable by such linear adaptations alone, potentially requiring more sophisticated, non-linear mapping techniques within the latent space.

Linear complexity with respect to augmented views. Finally, although STS is notably more lightweight than current state-of-the-art TTA strategies, its computational demand for visual processing scales linearly with the number of augmented views due to the need for independent forward passes. An intriguing future direction is to explore latent visual space augmentation to eliminate these repeated encoder computations.

By addressing these current limitations, the robustness and performance of spectrum-aware test-time adaptation strategies could be further advanced.

7 Conclusion

We propose Spectrum-Aware Test-Time Steering (STS), a lightweight adaptation framework for vision-language models like CLIP. STS exploits the spectral structure of text embeddings to define a compact semantic subspace, where it learns a per-sample steering vector to improve zero-shot robustness without modifying the frozen encoders. Our experiments show that STS consistently enhances performance across diverse benchmarks, offering an efficient and practical test-time adaptation method.

Notably, in addition to the text prototypes, our proposed STS method can be readily applied to the visual embeddings as well. Exploring under which conditions and settings the STS should be preferred over text prototypes or visual embeddings constitutes an interesting research direction that belongs to our future agenda.

Acknowledgments

We thank Konstantinos D. Polyzos for insightful discussions and for his helpful and detailed comments. We also thank the reviewers for their constructive suggestions, which have helped improve the quality of this paper. This research has been partially funded by research grants to D. Metaxas through NSF: 2310966, 2235405, 2212301, 2003874, 1951890, AFOSR 23RT0630, and NIH 2R01HL127661.

References

- [1] Jameel Abdul Samadh, Mohammad Hanan Gani, Noor Hussein, Muhammad Uzair Khattak, Muhammad Muzammal Naseer, Fahad Shahbaz Khan, and Salman H Khan. “Align your prompts: Test-time prompting with distribution alignment for zero-shot generalization”. In: *Advances in Neural Information Processing Systems* 36 (2023), pp. 80396–80413.
- [2] Armen Aghajanyan, Luke Zettlemoyer, and Sonal Gupta. “Intrinsic dimensionality explains the effectiveness of language model fine-tuning”. In: *arXiv preprint arXiv:2012.13255* (2020).
- [3] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. “Food-101—mining discriminative components with random forests”. In: *Computer vision—ECCV 2014: 13th European conference, zurich, Switzerland, September 6–12, 2014, proceedings, part VI* 13. Springer. 2014, pp. 446–461.
- [4] Evgenia Tatiani Chroni, Konstantinos M Dafnis, Georgios Chantzialexiou, Eric Cosatto, and Dimitris Metaxas. “Improving Test-Time Adaptation For Histopathology Image Segmentation: Gradient-To-Parameter Ratio Guided Feature Alignment”. In: *2024 IEEE International Symposium on Biomedical Imaging (ISBI)*. IEEE. 2024, pp. 1–5.
- [5] Mircea Cimpoi, Subhansu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. “Describing textures in the wild”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2014, pp. 3606–3613.
- [6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. “Imagenet: A large-scale hierarchical image database”. In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255.
- [7] Matteo Farina, Gianni Franchi, Giovanni Iacca, Massimiliano Mancini, and Elisa Ricci. “Frustratingly easy test-time adaptation of vision-language models”. In: *arXiv preprint arXiv:2405.18330* (2024).
- [8] Li Fei-Fei, Rob Fergus, and Pietro Perona. “Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories”. In: *2004 conference on computer vision and pattern recognition workshop*. IEEE. 2004, pp. 178–178.
- [9] Chun-Mei Feng, Kai Yu, Yong Liu, Salman Khan, and Wangmeng Zuo. “Diverse data augmentation with diffusions for effective test-time prompt tuning”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pp. 2704–2714.
- [10] Peng Gao, Shijie Geng, Renrui Zhang, Teli Ma, Rongyao Fang, Yongfeng Zhang, Hongsheng Li, and Yu Qiao. “Clip-adapter: Better vision-language models with feature adapters”. In: *International Journal of Computer Vision* 132.2 (2024), pp. 581–595.
- [11] Yunhe Gao, Xingjian Shi, Yi Zhu, Hao Wang, Zhiqiang Tang, Xiong Zhou, Mu Li, and Dimitris N Metaxas. “Visual prompt tuning for test-time domain adaptation”. In: *arXiv preprint arXiv:2210.04831* (2022).
- [12] Matan Gavish and David L Donoho. “Optimal shrinkage of singular values”. In: *IEEE Transactions on Information Theory* 63.4 (2017), pp. 2137–2152.
- [13] Patrick Helber, Benjamin Bischke, Andreas Dengel, and Damian Borth. “Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification”. In: *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* 12.7 (2019), pp. 2217–2226.
- [14] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. “The many faces of robustness: A critical analysis of out-of-distribution generalization”. In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2021, pp. 8340–8349.
- [15] Dan Hendrycks and Thomas Dietterich. “Benchmarking neural network robustness to common corruptions and perturbations”. In: *arXiv preprint arXiv:1903.12261* (2019).

- [16] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. “Natural adversarial examples”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2021, pp. 15262–15271.
- [17] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. “Lora: Low-rank adaptation of large language models.” In: *ICLR* 1.2 (2022), p. 3.
- [18] Raza Imam, Hanan Gani, Muhammad Huzaifa, and Karthik Nandakumar. “Test-time low rank adaptation via confidence maximization for zero-shot generalization of vision-language models”. In: *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. IEEE. 2025, pp. 5449–5459.
- [19] Chao Jia, Yinfei Yang, Ye Xia, Yi-Ting Chen, Zarana Parekh, Hieu Pham, Quoc Le, Yun-Hsuan Sung, Zhen Li, and Tom Duerig. “Scaling up visual and vision-language representation learning with noisy text supervision”. In: *International conference on machine learning*. PMLR. 2021, pp. 4904–4916.
- [20] Adilbek Karmanov, Dayan Guan, Shijian Lu, Abdulmotaleb El Saddik, and Eric Xing. “Efficient test-time adaptation of vision-language models”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2024, pp. 14162–14171.
- [21] Muhammad Uzair Khattak, Hanoona Rasheed, Muhammad Maaz, Salman Khan, and Fahad Shahbaz Khan. “Maple: Multi-modal prompt learning”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2023, pp. 19113–19122.
- [22] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. “3d object representations for fine-grained categorization”. In: *Proceedings of the IEEE international conference on computer vision workshops*. 2013, pp. 554–561.
- [23] Yuejiang Liu, Parth Kothari, Bastien Van Delft, Baptiste Bellot-Gurlet, Taylor Mordan, and Alexandre Alahi. “Ttt++: When does self-supervised test-time training fail or thrive?” In: *Advances in Neural Information Processing Systems* 34 (2021), pp. 21808–21820.
- [24] Ilya Loshchilov and Frank Hutter. “Decoupled weight decay regularization”. In: *arXiv preprint arXiv:1711.05101* (2017).
- [25] Subhransu Maji, Esa Rahtu, Juho Kannala, Matthew Blaschko, and Andrea Vedaldi. “Fine-grained visual classification of aircraft”. In: *arXiv preprint arXiv:1306.5151* (2013).
- [26] Muhammad Jehanzeb Mirza, Pol Jané Soneira, Wei Lin, Mateusz Kozinski, Horst Possegger, and Horst Bischof. “Actmad: Activation matching to align distributions for test-time-training”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 24152–24161.
- [27] Maria-Elena Nilsback and Andrew Zisserman. “Automated flower classification over a large number of classes”. In: *2008 Sixth Indian conference on computer vision, graphics & image processing*. IEEE. 2008, pp. 722–729.
- [28] OpenAI. *CLIP: Contrastive Language-Image Pretraining*. <https://github.com/openai/CLIP>. 2021.
- [29] Omkar M Parkhi, Andrea Vedaldi, Andrew Zisserman, and CV Jawahar. “Cats and dogs”. In: *2012 IEEE conference on computer vision and pattern recognition*. IEEE. 2012, pp. 3498–3505.
- [30] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. “Learning transferable visual models from natural language supervision”. In: *International conference on machine learning*. PmLR. 2021, pp. 8748–8763.
- [31] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. “Do imagenet classifiers generalize to imagenet?” In: *International conference on machine learning*. PMLR. 2019, pp. 5389–5400.
- [32] Manli Shu, Weili Nie, De-An Huang, Zhiding Yu, Tom Goldstein, Anima Anandkumar, and Chaowei Xiao. “Test-time prompt tuning for zero-shot generalization in vision-language models”. In: *Advances in Neural Information Processing Systems* 35 (2022), pp. 14274–14289.
- [33] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. “UCF101: A dataset of 101 human actions classes from videos in the wild”. In: *arXiv preprint arXiv:1212.0402* (2012).
- [34] Elaine Sui, Xiaohan Wang, and Serena Yeung-Levy. “Just shift it: Test-time prototype shifting for zero-shot generalization with vision-language models”. In: *arXiv preprint arXiv:2403.12952* (2024).

- [35] Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei Efros, and Moritz Hardt. “Test-time training with self-supervision for generalization under distribution shifts”. In: *International conference on machine learning*. PMLR. 2020, pp. 9229–9248.
- [36] Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. “Tent: Fully test-time adaptation by entropy minimization”. In: *arXiv preprint arXiv:2006.10726* (2020).
- [37] Haohan Wang, Songwei Ge, Zachary Lipton, and Eric P Xing. “Learning robust global representations by penalizing local predictive power”. In: *Advances in neural information processing systems* 32 (2019).
- [38] Jianxiong Xiao, James Hays, Krista A Ehinger, Aude Oliva, and Antonio Torralba. “Sun database: Large-scale scene recognition from abbey to zoo”. In: *2010 IEEE computer society conference on computer vision and pattern recognition*. IEEE. 2010, pp. 3485–3492.
- [39] Hee Suk Yoon, Eunseop Yoon, Joshua Tian Jin Tee, Mark Hasegawa-Johnson, Yingzhen Li, and Chang D Yoo. “C-tp: Calibrated test-time prompt tuning for vision-language models via text feature dispersion”. In: *arXiv preprint arXiv:2403.14119* (2024).
- [40] Renrui Zhang, Wei Zhang, Rongyao Fang, Peng Gao, Kunchang Li, Jifeng Dai, Yu Qiao, and Hongsheng Li. “Tip-adapter: Training-free adaption of clip for few-shot classification”. In: *European conference on computer vision*. Springer. 2022, pp. 493–510.
- [41] Yabin Zhang, Wenjie Zhu, Hui Tang, Zhiyuan Ma, Kaiyang Zhou, and Lei Zhang. “Dual memory networks: A versatile adaptation approach for vision-language models”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2024, pp. 28718–28728.
- [42] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. “Conditional prompt learning for vision-language models”. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2022, pp. 16816–16825.
- [43] Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. “Learning to prompt for vision-language models”. In: *International Journal of Computer Vision* 130.9 (2022), pp. 2337–2348.

Test-Time Spectrum-Aware Latent Steering for Zero-Shot Generalization in Vision-Language Models

Appendix

In this supplementary document, we provide additional details and experimental results to enhance understanding and insights into our method. This supplementary document is organized as follows:

- Broader Impact of our method in Section A.
- We present an error bar analysis for the results in Table 1 and Table 2 in the main document, in Section B.1.
- We provide an analysis on the impact of varying update steps in Section B.2.
- We evaluate the effect of shared versus per-class coefficients vector in Section B.3.
- We present additional performance comparisons on larger-scale VLMs, specifically OpenAI CLIP [30] with a ViT-L/14 backbone, in Section B.4.
- We analyze the effect of the singular vector selection for the test-time latent steering, in Section C.
- We provide the detailed statistics for all the utilized datasets, and the specific textual prompts that we used in Section D.
- We list the license information for all used assets in Section E.

A Broader Impact

This research contributes to the overarching goal of developing more dependable and effective machine learning systems by enabling large foundation models like CLIP [30] to dynamically adapt to real-world operational conditions at test time. Such adaptability is critical for deploying AI robustly in diverse and unpredictable environments, thereby broadening their practical applications and fostering greater system reliability. Ultimately, we hope this work stimulates and guides future studies focused on enhancing the generalization capabilities and operational robustness of pre-trained models, ensuring they can be utilized more effectively and responsibly to address a wide array of societal challenges.

B Technical Appendices

B.1 Analysis on error bars

We run STS multiple times using 3 different random seeds and report the average accuracy with standard deviation in Table B1. The randomness of STS mainly comes from random data augmentation. Our augmentation setup is simple and only contains resized random crops and random horizontal flips, which can constitute a “zoom-in” to a random portion of the image. We did not search for the best data augmentations, but rather stuck to an established setting, using the same augmentation setup for all datasets. However, the performance of STS is linked to the impact that data augmentations have on how the model perceives images, and we believe that this is an interesting research direction to pursue. In addition, we report an error bar analysis for the results in Table 1 and Table 2 in the main document.

Table B1: Robustness to natural distribution shifts. We report the accuracy with an error bar (standard deviation) obtained from three runs with different random seeds. The best results in each section are highlighted in **bold**. Underline indicates second-best.

Method	ImageNet	A	V2	R	Sketch	Average	OOD Average
CLIP-ViT-B/16							
TPT [32]	68.97 (± 0.04)	54.39 (± 0.20)	63.37 (± 0.06)	77.07 (± 0.06)	48.01 (± 0.08)	62.36 (± 0.03)	60.71 (± 0.04)
C-TPT [39]	68.53 (± 0.02)	51.14 (± 0.09)	62.13 (± 0.11)	75.66 (± 0.07)	47.37 (± 0.08)	60.97 (± 0.02)	59.08 (± 0.02)
TPS [34]	67.96 (± 0.02)	57.46 (± 0.12)	62.95 (± 0.11)	74.90 (± 0.04)	46.03 (± 0.09)	61.86 (± 0.06)	60.34 (± 0.07)
STS (Ours)	68.85 (± 0.03)	<u>61.23</u> (± 0.26)	64.15 (± 0.20)	<u>77.13</u> (± 0.06)	<u>48.06</u> (± 0.06)	<u>63.88</u> (± 0.08)	<u>62.64</u> (± 0.10)
STS _{Ensemble}	70.81 (± 0.04)	64.29 (± 0.09)	<u>64.82</u> (± 0.14)	80.53 (± 0.13)	50.19 (± 0.02)	66.13 (± 0.01)	64.96 (± 0.005)

Table B2: Performance comparisons on cross-dataset generalization from ImageNet to fine-grained classification datasets. We report the accuracy with an error bar (standard deviation) obtained from three runs with different random seeds. The best results in each section are highlighted in **bold**. Underline indicates second-best.

Method	Flowers102	DTD	OxfordPets	UCF101	Caltech101	Aircraft	EuroSAT	StanfordCars	Food101	SUN397	Average
CLIP-ViT-B/16											
TPT [32]	68.98 (± 0.13)	47.16 (± 0.08)	87.07 (± 0.19)	67.89 (± 0.07)	<u>94.19</u> (± 0.12)	22.85 (± 0.41)	<u>43.01</u> (± 0.06)	66.55 (± 0.02)	84.67 (± 0.06)	<u>65.47</u> (± 0.13)	<u>64.78</u> (± 0.05)
C-TPT [39]	69.88 (± 0.19)	45.54 (± 0.13)	87.96 (± 0.14)	65.19 (± 0.41)	93.39 (± 0.16)	24.13 (± 0.24)	38.43 (± 0.45)	65.26 (± 0.24)	82.60 (± 0.16)	63.38 (± 0.26)	63.58 (± 0.05)
TTL [18]	67.32 (± 0.25)	45.92 (± 0.02)	86.78 (± 0.02)	<u>67.80</u> (± 0.06)	93.23 (± 0.06)	24.00 (± 0.36)	36.52 (± 0.05)	65.95 (± 0.24)	84.40 (± 0.02)	64.02 (± 0.05)	63.59 (± 0.02)
TPS [34]	66.14 (± 0.11)	45.49 (± 0.36)	86.56 (± 0.08)	66.53 (± 0.19)	93.60 (± 0.12)	24.01 (± 0.44)	37.85 (± 0.23)	66.93 (± 0.18)	82.96 (± 0.09)	64.85 (± 0.07)	63.49 (± 0.07)
STS (Ours)	66.10 (± 0.15)	46.02 (± 0.10)	86.69 (± 0.18)	66.52 (± 0.10)	93.72 (± 0.10)	24.57 (± 0.06)	38.26 (± 0.35)	<u>67.17</u> (± 0.27)	<u>84.72</u> (± 0.06)	64.79 (± 0.17)	63.86 (± 0.06)
STS _{Ensemble}	67.16 (± 0.32)	<u>46.87</u> (± 0.12)	<u>87.11</u> (± 0.08)	67.14 (± 0.06)	94.20 (± 0.05)	<u>24.21</u> (± 0.05)	43.80 (± 0.15)	68.16 (± 0.23)	85.15 (± 0.03)	66.79 (± 0.04)	65.06 (± 0.03)

B.2 Impact of Varying Update Steps

By default, STS updates the coefficients using a single step per test instance. The optimal learning rate for this single step is determined to be 0.005 on the standard ImageNet validation set (not including any of the out-of-distribution data). To evaluate the impact of different numbers of update steps on overall performance, we conduct ablation experiments by varying the number of update steps from 1 to 5 and report the resulting performance on ImageNet-A. For these multi-step ablations, the value of λ_R of the regularization loss is set to 0.01, and the initial learning rate remains 0.005. Since it is optimal for a single update step, all subsequent steps are subjected to a learning rate schedule, applying a one-time decay factor of 0.1 to the initial rate. As shown in Table B3, the number of update steps does not significantly influence performance (in the range of 0.1%). Although increasing the update steps to 2 yields a slight performance gain of 0.02%, it also leads to a proportional decrease in inference efficiency. Although our method is extremely efficient, given this trade-off, we adopt the single-step update as the default for balancing efficiency and performance.

Table B3: Ablation study on different update steps in learning the steering vectors. We vary the number of update steps from 1 to 5 and report the achieved performance on ImageNet-A. Results are over 3 random seeds.

# Steps	1	2	3	4	5
Accuracy	61.23	61.25	61.22	61.20	61.20

B.3 Effect of Shared vs. Per-Class Coefficients

Test-time adaptation strategies vary in how they modify class representations. For instance, prompt tuning methods adjust a shared prompt that subsequently undergoes non-linear transformations via the text encoder. In the context of latent-space adaptation, class prototypes can be modulated either by individual, per-class coefficients or by a single, shared coefficients vector. A shared vector applies a uniform transformation, thus maintaining the relative geometric structure of the class prototypes after adaptation. This approach primarily targets global, dataset-level distribution shifts. Although per-class vectors could, in principle, offer finer control by providing more degrees of freedom to capture distinct class-level shifts within a domain gap, the practical benefits of such granularity warrant careful consideration. Our work investigates the efficacy of the shared coefficients vector

approach, as the additional complexity introduced by per-class vectors may not consistently translate into substantial performance gains over a simpler, unified shift.

As shown in Table B4, learning a per-class shift yields a marginal average performance increase of only 0.03%. Similarly, Table B5 indicates that per-class coefficients provide a mere 0.01% average improvement in cross-dataset generalization from ImageNet to fine-grained classification tasks. These minimal gains suggest that, at least for a single update step, learning per-class coefficients does not substantially enhance model performance when encountering a domain gap.

Table B4: A performance comparison of shared versus per-class steering vectors regarding robustness to natural distribution shifts. We present the average top-1 accuracy (%) results over 3 random seeds for a single update step. The best performance is highlighted in **bold**.

Method	ImageNet	A	V2	R	Sketch	Average	OOD Average
CLIP-ViT-B/16							
Shared	68.85	61.23	64.15	77.13	48.06	63.88	62.64
Per-class	68.91	61.24	64.20	77.15	48.06	63.91	62.66

Table B5: A performance comparison of shared versus per-class steering vectors on cross-dataset generalization from ImageNet to fine-grained classification datasets. We present the average top-1 accuracy (%) results over 3 random seeds for a single update step. The best performance is highlighted in **bold**.

Method	Flowers102	DTD	OxfordPets	UCF101	Caltech101	Aircraft	EuroSAT	StanfordCars	Food101	SUN397	Average
CLIP-ViT-B/16											
Shared	66.10	46.02	86.69	66.52	93.72	24.57	38.26	67.17	84.72	64.79	63.86
Per-class	66.15	45.88	86.71	66.46	93.78	24.60	38.23	67.27	84.77	64.83	63.87

B.4 Performance Comparison on Larger-Scale VLMs

Our STS method can theoretically be applied to various contrastively pre-trained vision-language models such as CLIP ViT-B/16 and CLIP ViT-L/14. In Table B6, we use OpenAI CLIP ViT-L/14 larger-scale OpenAI CLIP model, as an example, and compare the performance of our STS method and zero-shot on robustness to natural distribution shifts. We can observe that our STS still outperforms zero-shot by a large margin on average across 5 datasets, showcasing that our method generalizes well to larger-scale VLMs.

Table B6: Performance comparison on robustness to natural distribution shifts. We present top-1 accuracy (%) results by employing the larger-scale ViT-L/14 variant of CLIP [30]. The reported results of STS are based on a single random seed. The best performance is highlighted in **bold**.

Method	ImageNet	A	V2	R	Sketch	Average	OOD Average
CLIP-ViT-L/14							
Zero-Shot	73.45	68.76	67.79	85.39	57.81	70.64	69.94
STS (Ours)	75.37	78.52	69.88	88.07	59.85	74.34	74.08
Δ	+1.92	+9.76	+2.09	+2.68	+2.04	+3.70	+4.14

C Singular Vector Selection for Test-Time Latent Steering

In our test-time adaptation (TTA) approach, Singular Value Decomposition (SVD) is applied to text prototypes (e.g., “a photo of a [CLASS]”) to analyze their underlying semantic structure. The full set of singular vectors describes this structure completely. However, when adapting to a new out-of-distribution (OOD) domain at test time, not all components of this original structure may be equally relevant or contribute equally to successful adaptation. The process of selecting a subset of

singular vectors aims to distill the dimensions most pertinent for characterizing the new domain’s relationship to the class concepts, potentially leading to a more focused and effective adaptation. As indicated in the main document (Section 3), a substantial portion of the variance (e.g., 90%) is often concentrated in a significantly smaller subset of singular values, highlighting the potential for effective dimensionality reduction. Discarding low-variance components here means removing directions where our specific classes are textually very similar.

This section details two principled methods for selecting k_t singular vectors and presents their impact on the zero-shot performance of CLIP [30] when integrated with our TTA method. Our findings indicate that strategic selection of singular vectors significantly enhances performance, with the Gavish-Donoho method yielding slightly superior results.

C.1 Energy-Based Singular Vector Selection

A common heuristic for dimensionality reduction via SVD is to retain the top- k_t singular values, $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_{k'}$, such that they capture a predefined percentage of the total "energy" (sum of squared singular values). We investigate a threshold of 98% energy, selecting k_t such that:

$$\frac{\sum_{i=1}^{k_t} \sigma_i^2}{\sum_{j=1}^{k'} \sigma_j^2} \geq 0.98$$

This method aims to preserve the most dominant components of variance within the text prototype manifold, assuming these capture the most salient semantic information. Based on our experiments $k_{98\%}$ is typically less than k' .

C.2 Gavish-Donoho Optimal Hard Thresholding

The Gavish-Donoho method [12] offers a theoretically grounded approach for selecting an optimal number of singular values k_t , to retain, particularly when seeking a robust low-rank representation of the data. Developed from random matrix theory, this method computes an optimal singular value threshold ω^* . This threshold is designed to effectively separate the more dominant and structured components within the singular value spectrum from those that are less influential or exhibit characteristics similar to the singular values of a random matrix. The singular values $\sigma_i < \omega^*$ (and their corresponding singular vectors) are consequently excluded, leading to the determination of the rank k_t . The specific threshold value depends on the aspect ratio of the matrix undergoing SVD and can be established using the properties of the singular value spectrum itself (e.g., via the median singular value), thus providing a data-driven cutoff without requiring an explicit "noise" model. We apply this method to determine k_t for our text prototype matrix, in order to identify a subset of singular vectors that is the basis of adaptation.

C.3 Performance Impact of Singular Vector Selection

To demonstrate the efficacy of these selection strategies, we evaluate our TTA method on ImageNet-A dataset. Table C1 presents the top-1 accuracy, comparing the zero-shot CLIP baseline [30] with our TTA approach when using singular vectors selected by the 98% energy criterion ($k_{98\%}$) versus the Gavish-Donoho method (k_t).

Table C1: Impact of singular vector selection on Test-Time Adaptation (TTA) performance. Average Top-1 accuracy (%) over 3 random seeds is reported. Both selection methods significantly improve over the zero-shot baseline, with Gavish-Donoho (GD) offering a slight further advantage.

Dataset	Zero-Shot (ViT-B/16)	STS (Ours) ($k_{98\%}$ SVs)	STS (Ours) (k_t SVs)
ImageNet-A [16]	47.87	61.09 (+13.22)	61.23 (+13.36)

The results in Table C1 clearly indicate that employing a principled selection of singular vectors substantially boosts the performance of our TTA method compared to the zero-shot baseline. Both the 98% energy criterion and the Gavish-Donoho threshold lead to significant improvements. Notably, the Gavish-Donoho method consistently achieves slightly better performance, suggesting its

effectiveness in identifying an optimal rank for the text prototype subspace used in our adaptation process. This underscores the importance of focusing the adaptation on the most informative and semantic dimensions derived from the text prototypes.

D Additional Implementation Details

D.1 Dataset Details

In Table D1, we present the detailed statistics of each dataset we used in our experiments, including the number of classes, the sizes of training, validation and testing sets, and their original tasks.

Table D1: Detailed statistics of datasets used in experiments. Note that the 4 ImageNet variant datasets are designed for evaluation and only contain the test sets.

Dataset	Classes	Training	Validation	Testing	Task
ImageNet [6]	1,000	1.28M	-	50,000	Object recognition
ImageNet-A [16]	200	-	-	7,500	Robustness of adversarial attack
ImageNet-V2 [31]	1,000	-	-	10,000	Robustness of collocation
ImageNet-R [14]	200	-	-	30,000	Robustness of multi-domains
ImageNet-Sketch [37]	1,000	-	-	50,889	Robustness of sketch domain
Caltech101 [8]	100	4,128	1,649	2,465	Object recognition
DTD [5]	47	2,820	1,128	1,692	Texture recognition
EuroSAT [13]	10	13,500	5,400	8,100	Satellite image recognition
FGVCAircraft [25]	100	3,334	3,333	3,333	Fine-grained aircraft recognition
Flowers102 [27]	102	4,093	1,633	2,463	Fine-grained flowers recognition
Food101 [3]	101	50,500	20,200	30,300	Fine-grained food recognition
OxfordPets [29]	37	2,944	736	3,669	Fine-grained pets recognition
StanfordCars [22]	196	6,509	1,635	8,041	Fine-grained car recognition
SUN397 [38]	397	15,880	3,970	19,850	Scene recognition
UCF101 [33]	101	7,639	1,898	3,783	Action recognition

In Table D2, we detail the specific hand-crafted prompts utilized in our experiments.

Table D2: Datasets with associated textual prompts. The first prompt is applied generally, while the subsequent *generic* prompts (indicated by the brace) are collectively used as an ensemble for each dataset. These 7 generic templates are highlighted in the official CLIP repository [28].

Dataset	Prompts
ImageNet [6]	“a photo of a {CLASS}.” “a bad photo of the {CLASS}.” “a {CLASS} in a video game.” “a origami {CLASS}.” “a photo of the small {CLASS}.” “art of the {CLASS}.” “a photo of the large {CLASS}.” “itap of a {CLASS}.”
ImageNet-V2 [31]	
ImageNet-Sketch [37]	
ImageNet-A [16]	
ImageNet-R [14]	
Caltech101 [8]	
DTD [5]	
EuroSAT [13]	“a photo of the small {CLASS}.” “art of the {CLASS}.” “a photo of the large {CLASS}.” “itap of a {CLASS}.”
FGVCAircraft [25]	
Flowers102 [27]	
Food101 [3]	
OxfordPets [29]	
StanfordCars [22]	
SUN397 [38]	
UCF101 [33]	

→ Ensemble

E License Information

Datasets. We list the known license information for the datasets below:

- CC BY-SA 4.0 License: OxfordPets [29].
- MIT License: ImageNet-A [16], ImageNet-V2 [31], ImageNet-R [14], and ImageNet-Sketch [37].
- Research Purposes only (term of access): ImageNet [6], DTD [5], StanfordCars [22], SUN397 [38], FGVC Aircraft [25].

Source Code. In this work, we also use some code implementations from existing baseline methods to report their results: CLIP [30], CoOp [43], MaPLe [21], TPT [32]. The source code used in this paper for these methods is available under the MIT License.

NeurIPS Paper Checklist

1. Claims

Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

Answer: [\[Yes\]](#)

Justification: We clearly state our contributions in Abstract and also in the first Section.

Guidelines:

- The answer NA means that the abstract and introduction do not include the claims made in the paper.
- The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
- The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
- It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. Limitations

Question: Does the paper discuss the limitations of the work performed by the authors?

Answer: [\[Yes\]](#)

Justification: We have discussed the limitations of the work in Section 6.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. Theory assumptions and proofs

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [\[NA\]](#)

Justification: This paper does not include theoretical results.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. Experimental result reproducibility

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [\[Yes\]](#)

Justification: We believe that we have clearly introduced the proposed STS method in the main text for reproduction. We have provided all the training details, and computational resources.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general, releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
 - (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
 - (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
 - (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
 - (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. Open access to data and code

Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No]

Justification: We plan to release the code after the decision.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so “No” is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (<https://nips.cc/public/guides/CodeSubmissionPolicy>) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. Experimental setting/details

Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We followed all the standard benchmarks for our experiments following previous works.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
- The full details can be provided either with the code, in appendix, or as supplemental material.

7. Experiment statistical significance

Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [No]

Justification: We plan to report error bars in the supplemental material.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).

- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. Experiments compute resources

Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: We provide all the details in the Section of Experiments.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. Code of ethics

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics <https://neurips.cc/public/EthicsGuidelines>?

Answer: [Yes]

Justification: We believe that the research conducted in the paper conform, is in every respect, with the NeurIPS Code of Ethics.

Guidelines:

- The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. Broader impacts

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [NA]

Justification: We do not believe that this work has a negative social impact.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.

- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. Safeguards

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: The paper poses no such risks.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. Licenses for existing assets

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: Yes, we will report them in the supplemental material.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, paperswithcode.com/datasets has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.

- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: This paper does not introduce new assets.

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification:his paper does not involve crowdsourcing or research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: This paper does not involve crowdsourcing or research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. Declaration of LLM usage

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA]

Justification: We did not use LLMs.

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (<https://neurips.cc/Conferences/2025/LLM>) for what should or should not be described.